REMARKS

Applicant thanks the examiner for the examiner's time and helpful comments during the January 16, 2007, telephone interview between the applicant's undersigned attorney and the examiner. No immediate agreement was reached.

Claims 1-3, 5 and 7-22 are pending. Claims 1, 10, 16 and 19 are independent.

The examiner rejected independent claims 1, 10, 16 and 19 under 35 U.S.C. §101. Specifically, the examiner stated in the December 22, 2006, Office Action:

> 3. As to claims 1,19, claims 1, 19 are not limited to tangible embodiment, in view of Applicant's disclosure, specification page 2, lines 6-13, the computer program product is not limited to tangible embodiments, instead being defined as including both tangible embodiments (e.g., (microprocessor)) and intangible embodiment (e.g. (internet). Although applicant amended the program product residing on a computer readable medium for causing the execution, no practical application can be found in the clamed invention. The examiner understands the program product being stored in the computer readable medium to cause execution , but the focus is not on whether the steps taken to achieve a particular result is useful, tangible, concrete, but rather the final result is useful, tangible and concrete (see newly updated MPEP 2106, 2100-10-12). Claim 1 recites to cause execution, and it is read as a step taken to achieve practical result, but it is not a final result. Claim 1 also recites to branch to an instruction at a specific address at a value of availability of resource, but no substantial practical application can be found in the claim. As such, the claim is not useful, tangible and concrete, and is therefore non-statutory.
> 4. As to applicant's remark that claims 1 and 19 achieve a final result determination of how program flow of the currently executing instruction sequence of instruction stream is to proceed, the determination of how the program would go is an intended result, not a positive limitation. (Office Action, pages 2-3, paragraphs 3-4)

Applicant traverses the examiner's rejections of independent claims 1, 10, 16 and 19 under 35 U.S.C. §101.

Applicant amended independent claim 1 and 19 to replace the wording "medium" with "device", thus clarifying that the subject matter of independent claims 1 and 19 is directed to tangible embodiments of the claimed computer program product (that is, computer program product stored on a computer-readable storage device).

As for the examiner's contention that the subject-matter recited in these claims allegedly does not achieve a useful, tangible and concrete final result, applicant respectfully disagrees.

As stated in MPEP 2106.IV.C.2(2):

> **(2) Practical Application That Produces a Useful, Concrete, and Tangible Result**
>
> ...
>
> If USPTO personnel determine that the claim does not entail the transformation of an article, then USPTO personnel shall review the claim to determine it produces a useful, tangible, and concrete result. In making this determination, the focus is not on whether the steps taken to achieve a particular result are useful, tangible, and concrete, but rather on whether the final result achieved by the claimed invention is "useful, tangible, and concrete." In other words, the claim must be examined to see if it includes anything more than a 35 U.S.C. 101 judicial exception. If the claim is directed to a practical application of a 35 U.S.C. 101 judicial exception, USPTO personnel must then determine whether the claim preempts the judicial exception. If USPTO personnel do not find such a practical application, then USPTO personnel have determined that the claim is nonstatutory.
>
> In determining whether a claim provides a practical application of a 35 U.S.C. 101 judicial exception that produces a useful, tangible, and concrete result, USPTO personnel should consider and weigh the following factors:
>
> a) **"USEFUL RESULT"**
>
> For an invention to be "useful" it must satisfy the utility requirement of section 101. The USPTO's official interpretation of the utility requirement provides that the utility of an invention has to be (i) specific, (ii) substantial and (iii) credible.
>
> ...
>
> b) **"TANGIBLE RESULT"**
>
> The tangible requirement does not necessarily mean that a claim must either be tied to a particular machine or apparatus or must operate to change articles or materials to a different state or thing. However, the tangible requirement does require that the claim must recite more than a 35 U.S.C. 101 judicial exception, in that the process claim must set forth a practical application of that judicial exception to produce a real-world result.
>
> ...
>
> In other words, the opposite meaning of "tangible" is "abstract."
>
> c) **"CONCRETE RESULT"**
>
> Another consideration is whether the invention produces a "concrete" result. Usually, this question arises when a result cannot be assured. In other words, the process must have a result that can be substantially repeatable or the process must substantially produce the same result again.
>
> ... (emphasis added)

Applicant's independent claims 1 and 19 recite a branch instruction that causes a data processing apparatus and/or processor to evaluate a branch condition relating to whether a state, of a state name indicating the availability of a resource, is a specified value, and to branch to a specified address based on that evaluation. Thus, applicant's claims are all directed to branching operation that is useful, tangible and concrete.

Applicant's claims are tangible in that the claims are all tied to a particular machine (namely, a data processing apparatus or processor). Similarly Applicant's claims are all concrete, since the recited branching operation is repeatable and produces the same results upon different performances of the recited operation (e.g., branching will occur each time that the state of the state name specified in the branch instruction is a specified value). As such, applicant's claimed subject matter provides a concrete result as opposed to an abstract result.

Applicant further notes the recited branch operations are also final results in that the sequence of operations recited in the independent claims achieve a final determination of how program flow of the currently executing instruction sequence of instruction stream is to proceed.

With respect to independent claims 10 and 16, the examiner stated:

> 5. As to claim 10, no substantial practical application can be found in the claim. The practical application of the performance of the branch is not clear.
> 6. As to claim 16, although claim additionally recites the register stack and arithmetic unit coupled to the register stack, it is read as general arrangement of the system components, and present no substantial practical application. The practical application of the performance of the branch operation cannot be found. (Office Action, page 3, paragraphs 5-6)

In response, applicant amended independent claims 10 and 16 to recite the feature, similar to the feature recited in claim 1, that the branching operation causes an instruction stream to branch to another instruction of the instruction stream at an address specified in the branch instruction. Applicant made a similar amendment to claim 19. Applicant also notes that claim 1 was amended to replace "an instruction" with "another instruction of the instruction stream" for greater clarity.

Applicant contends that the examiner's rejections of independent claims 1, 10, 16 and 19 under 35 U.S.C. §101 is improper and/or overcome and thus should be withdrawn.

The examiner rejected claim 1 on the ground of non-statutory obviousness-type double patenting as being unpatentable over claim 29 of U.S. Patent No. 6,668,317. Specifically, the examiner stated:

> 9. Claim 1 rejected on the ground of nonstatutory obviousness-type double patenting as being unpatentable over claim 29 (claim 29 includes limitations of parent claim 23) of U.S. Patent No. 6,668,317. Although the conflicting claims are not identical, they are not patentably distinct from each other because although patented claim 29 did not recite the branch to the instruction at a specific address as claimed, patented claim 29 did disclose the determination of witch thread to execute based on signal indicating the completion of requested (see patented claim 23, lines 10-14). It would have been recognizable to one of ordinary skill in the art that the determination of the thread could be a sequence of instructions with the leading instruction at a specific memory location or a predefined address. Although not specifically recited, one of ordinary skill in the art should be able to recognize the applicability of branching or pointing to the instruction at the specified address as claimed based on the execution of the thread determination. (Office Action, pages 4-5, paragraph 9)

Applicant respectfully disagrees with the examiner's contentions. Applicant's independent claim 1 recites:

> 1. (Currently amended) A computer program product residing on a computer readable storage device and comprising instructions, including a branch instruction that when executed causes a data processing apparatus to:
>
> cause an executing instruction stream to branch to another instruction of the instruction stream at an address specified in the branch instruction if a state, of a state name specified in the branch instruction is a specified value, with the state indicating the availability of a resource of the data processing apparatus, wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads.

Applicant's independent claim 1 is directed to a branching operation, performed on a data processing apparatus that is based on the value of a state of a specified state name.

In contrast, claims 23 and 29 of U.S. Patent No. 6,668,317 respectively recite:

> 23. A network processor, comprising: at least one shared resource; and
>
> a collection of microcontrolled functional execution units communicatively coupled to the at least one shared resource, individual ones of the execution units to switch contexts between different threads executable in the execution unit, the threads to issue requests for operations on the at least one shared resource, the individual execution units including **a context event arbiter to determine which one of the threads to execute based, at least in part, on signals indicating when a thread's requested operation on the at least one shared resource completes**. (emphasis added)
>
> 29. The network processor of claim 28 in which the individual execution units further include an execution box data path having an arithmetic logic unit and a set of general purposes registers.

Thus, and as the examiner observed, claims 23 and 29 are directed to a network processor that can determine which thread to execute based on signals indicating when a thread's requested operation has completed. At no point, however, do claims 23 and 29, or for that matter any of the other claims of the '317 patent, recite subject matter related in anyway to a branching operation, which is an entirely different concept from switching contexts. Specifically, switching contexts, as understood by those skilled in the art, refers to causing a currently executing thread to stop execution and causing another thread to begin or resume execution. Branching, on the other hand, refers to jumping from one location in a currently executing instruction stream to another location in that same instruction stream.

Further, applicant contends that claims 23 and 29, or any of the other claims of the '317 patent, also do not recite that any type of operation is based on the value of a state of a specified state name. Rather, the decision to switch contexts, as recited in the claims of the '317 patent, is based on signals indicating when a thread's requested operation has completed. That decision is not based on the particular value of a state of a specified state name.

Accordingly, applicant contends that the subject matter recited in claims 23 and 29 in the '317 patent is entirely different and patentably distinct from the subject matter recited in applicant's independent claim 1. Applicant requests, therefore, that the non-statutory obviousness-type double patenting rejection of applicant's independent claim 1 be withdrawn.

The examiner rejected claims 1-20 under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 5,724,563 to Hasegawa in view of U.S. Patent No. 6,223,208 to Kiefer.

In addition, the examiner rejected claims 1 and 19 under 35 U.S.C. §103(a) as being unpatentable by U.S. Patent No. 4,454,595 to Cage in view of Kiefer. Further, the examiner rejected claims 1, 10 and 19 under 35 U.S.C. §103(a) as being unpatentable by U.S. Patent No. 6,275,508 to Aggarwal in view of Kiefer.

On the other hand, the examiner appears to have withdrawn his previously asserted rejections of claim 1 under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 5,640,538 to Dyer.

Specifically, with respect to the examiner's rejection of claim 1, as being obvious over Hasegawa in view of Kiefer, the examiner stated:

> 20. As to claims 1, Hasegawa disclosed a decisions on the direction of the instruction processing based on the state or condition (see also fig-s, and fig.10, see also the implicit value Z or C encoded in opcode in Table 1, col.1, lines 41-52, see the specific structure of also the teaching of Z and C flags set and reset in col.11, lines 14-40, lines 54-67, col.12, lines 1-5).
>
> 21. Hasegawa taught a data processing system including a branch instruction that caused an execution of instruction stream to branch to an instruction at an address (x) specified in the instruction if a state, of a specified name (Z,C), indicating the availability of a resource (see the branch instruction format in fig.2, see the value specified in the branch instruction field , see also fig.5, and fig.10, see also the implicit value Z or C encoded in opcode in Table 1 , col.11, lines 41-52, see also the teaching of Z and C flags set and reset in col.11 , lines 14-40, lines 54-67, col.12, lines 1-5). the parallel processor, see the pipeline processor in fig.6).
>
> 22. Hasegawa did not specifically show the process of a plurality of threads as claimed. However, Kiefer taught the microengine (see fig.2 [200]) having a control logic [context switching] and execution logic including the arithmetic unit [260]-[250] and general purpose unit [register set], and configured to process a plurality of threads (see the general purpose register and the execution of threads in col.8, lines 4-

> 48). It would have been obvious to one of ordinary skill in the art to use Kiefer in Hasegawa for including the plurality of threads as claimed because the use of Kiefer could increase the processing bandwidth of Hasegawa, and it could be done by predefining the plurality of threads into Hasegawa with modified configuration variables (e.g. thread length and thread type), and because Hasegawa taught a pipeline processing for processing a plurality of instructions in parallel by dividing the execution process into a plurality of processing stages (see col.1, lines 10-17), which was recognizable by one of ordinary skill in the art to use threads, or processing stages, as claimed in order to increase the processing bandwidth (e.g. the parallel processing), and in doing so, provided a motivation. (Office Action, pages 9-10, paragraphs 20-22)

Applicant respectfully disagrees with the examiner's contentions.

Applicant's independent claim 1 recites instructions to "cause an executing instruction stream to branch to another instruction of the instruction stream at an address specified in the branch instruction if a state, of a state name specified in the branch instruction is a specified value, with the state indicating the availability of a resource of the data processing apparatus, wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads."

Firstly, as applicant pointed out during the Examiner's Interview, the office action does not point out what part of the combined teachings of the Hasegawa and/or Kiefer the examiner relies upon as disclosing a plurality of microengines having the structure recited in independent claims 1, and/or the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," as required by applicant's independent claim 1. Accordingly, applicant submits that neither Hasegawa nor Kiefer discloses at least the above-mentioned features.

Hasegawa discloses a pipeline processor that can execute predictive branch instructions (Abstract). While Hasegawa's processor includes a register file 12 and an execution section 11, at no point does Hasegawa disclose that a plurality of such processors are used, each of which

being configured to process a plurality of threads. In fact, nowhere is it disclosed that Hasegawa's processor is capable of executing anything more than just a single process. Because Hasegawa's architecture does not include a plurality of microengines each configured to process a plurality of threads, Hasegawa's processor cannot make a state, indicating the availability of a resource, available to a plurality of microengines. Accordingly, contrary to the examiner's contention, Hasewaga does not disclose or suggest at least "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," as required by applicant's independent claim 1.

As for the examiner's contention that "[a]s to claim 4 [applicant notes that that claim has been previously cancelled], Hasegawa also included micro engines (see fig. 9 instruction register 8, instruction decoder 3 and execution unit 11)" (office action, page 10, paragraph 25), applicant notes that the Hasegawa disclosed components in FIG. 9 are not microengines for at least the reason that they do not each have "control logic and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," as required by independent claim 1.

Kiefer describes a multithreaded processor that uses idle register/storage functional units in a processor core to dynamically exchange threads (Kiefer, col. 1, lines 7-9). Specifically, Kiefre describes its processor core 200 as follows:

> **Returning now to FIG. 2, processor core 200 comprises at least an instruction register/control 230, a context switch unit 240, several register/register (RR) units 250, several register/storage (RS) units 260, and thread register sets 270, 272, 274. Inside each thread register set 270, 272, 274 are a number of specialized registers and components; the minimum being a general purpose register, one or more special purpose registers, a multiplier, a floating point register, and an arithmetic logic unit. Register/register units 250 and register/storage units 260 generally access the general purpose registers and the floating point registers from the executing thread register set 270. Register/register execution units 250 are operatively connected to receive instructions from the instruction register/control 230 and to transfer executed instruction results which do not require access to the memory system 220 into and out of the executing**

> **thread register set 270. Two register/register units 250 are illustrated in the embodiment shown in FIGS. 2 and 3, however, there can be as few as one or as many of these units as necessary or as defined by the computer architecture. Similarly, there can be few or numerous register/storage units 260 which perform instructions including those instructions that access the memory system 220. For the lazy context switching of the invention the register/storage units 260 read and write data from the thread register buffer sets 270, 272, and 274 and memory system 220, as enabled by the instruction register/control 230 and the context switch unit 240. Register/storage units 260 also receive normal instructions from the instruction register/control 230. Memory system 220 comprises the locations where the thread register states reside when not in the processor core 200. The state of a thread refers to the contents of the register files necessary for an instruction set capable of independent execution, called a thread, to execute. Of course, the state of a thread depends upon the processor architecture and implementation and, as an example only, a state can include the contents of a general purpose register file, a floating point register file, and other registers critical to the correct operation of the processor such as condition code registers, machine state registers, link and count registers, next instruction address register, exception registers, etc. (col. 8, lines 5-45)**

Nowhere does Kiefer describe a configuration that includes a plurality of processors such as processor 200. Further, because Kiefer does not describe a plurality of processors, it also doers not describe that a state of a specified state name is made available to such a plurality of processors. Accordingly, Kiefer also fails to disclose or suggest at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," as required by applicant's independent claim 1.

Because neither Hasegawa nor Kiefer discloses or suggests, alone or in combination, at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," applicant's independent claim 1 is therefore patentable over these prior art references.

As noted, the examiner also separately rejected independent claim 1 as being unpatentable over Cage in view of Kiefer, and also as being unpatentable over Aggarwal in view of Kiefer.

Cage describes a buffer for use with a word processing disk controller. Nowhere does Cage describe that any of the processors are multithreaded processors. Cage also does not describe that a state of a particular state name is made available to multithreaded processors. Cage also fails to disclose or suggest at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," as required by applicant's independent claim 1.

Aggarwal describes a system for processing datagram headers during traverses from one interface of a networking device to another (Abstract). Aggarwal's system includes a sequencer unit that controls the selection of data from input data stream (FIGS. 1 and 12, and Abstract). The sequencer unit is controlled by a word instruction called the Write Control Store (WCS) (see col. 3, lines 61-63).

Aggarwal, however, does not describe any processors, let alone multithreaded processors, and certainly does not describe a plurality of multithreaded processors. Aggarwal, therefore, also does not describe a state of a particular state name that is made available to a plurality of multithreaded processors. Accordingly, Aggarwal fails to disclose or suggest at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," as required by applicant's independent claim 1.

As explained above, Kiefer also does not disclose or suggest at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic and an execution box, the execution box including an arithmetic logic unit

and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads."

Because none of Aggarwal, Cage and Kiefer discloses or suggests, alone or in combination, at least the feature of "wherein the state is available to a plurality of microengines, each of the plurality microengines having control logic and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," applicant's independent claim 1 is therefore patentable over these prior art references.

Claims 2-3, 5, 7-9 and 21-22 depend from independent claim 1 and are therefore patentable for at least the same reasons as independent claim 1.

Independent claims 10, 16 and 19 recite "evaluating a value of a state name specified in a branch instruction, the value of the state name indicating the availability of a resource of the processor; and performing a branching operation to cause an instruction stream to branch to another instruction of the instruction stream at an address specified in the branch instruction based on the value of the specified state name being set or cleared, wherein the state is available to the multiple microengines, each of the multiple microengines having control logic and an execution box, the execution box including an arithmetic logic unit and a general purpose register set, each of the plurality of microengines being configured to process a plurality of threads," or similar language. At least these features are not disclosed by the prior art cited by the examiner for reasons similar to those provided with respect to independent claim 1. Accordingly, independent claims 10, 16, and 19 are patentable over the prior art.

Claims 11-15 depend from independent claim 10 and are therefore patentable for at least the same reasons as independent claim 10. Claims 17-18 depend from independent claim 16 and are therefore patentable for at least the same reasons as independent claim 16. Claim 20 depends from independent claim 19 and is therefore patentable for at least the same reasons as independent claim 19.

It is believed that all the rejections and/or objections raised by the examiner have been addressed.

In view of the foregoing, applicant respectfully submits that the application is in condition for allowance and such action is respectfully requested at the examiner's earliest convenience.

All of the dependent claims are patentable for at least the reasons for which the claims on which they depend are patentable.

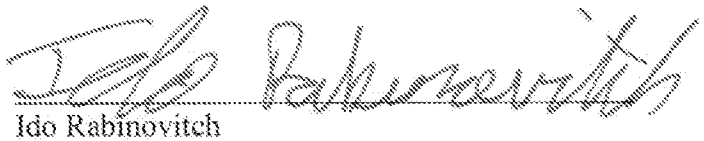
Canceled claims, if any, have been canceled without prejudice or disclaimer.

Any circumstance in which the applicant has (a) addressed certain comments of the examiner does not mean that the applicant concedes other comments of the examiner, (b) made arguments for the patentability of some claims does not mean that there are not other good reasons for patentability of those claims and other claims, or (c) amended or canceled a claim does not mean that the applicant concedes any of the examiner's positions with respect to that claim or other claims.

No fee is believed due. Please apply charges to deposit account 06-1050, referencing attorney docket 10559-306US1.

Respectfully submitted,

Date: Feb. 9, 2007



Ido Rabinovitch
Attorney for Intel Corporation
Reg. No. L0080

PTO Customer No. 20985
Fish & Richardson P.C.
Telephone: (617) 542-5070
Facsimile: (617) 542-8906